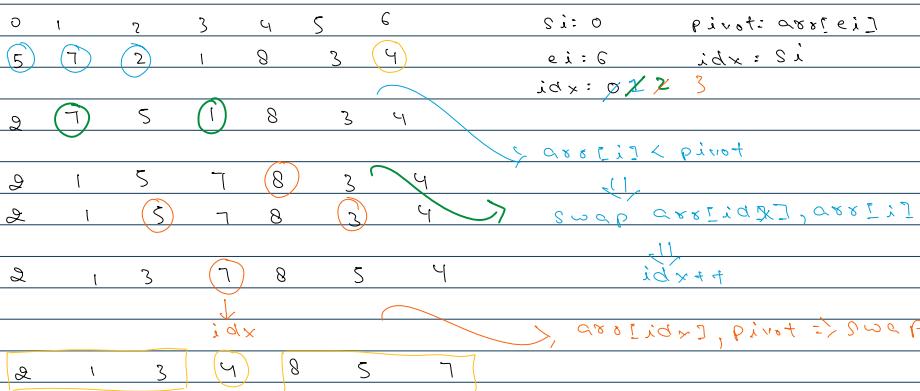
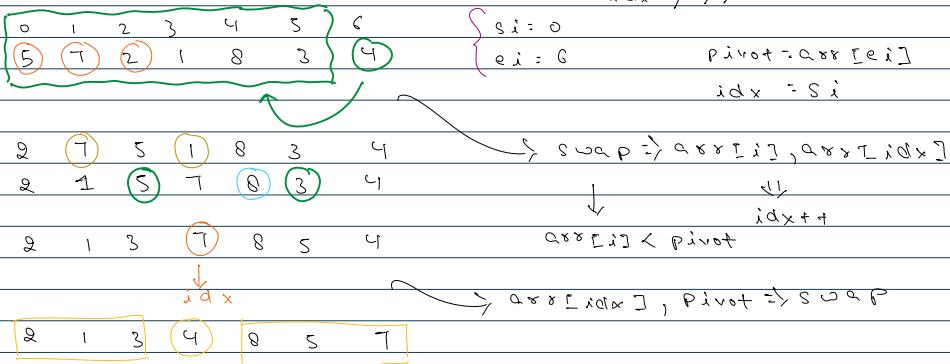


Partition in array

$n \log n \times$  } Do not sort  
the whole array  
number of possibilities,



$idx = 0 \cancel{\neq} 3$



```
public class Main {
    public static void main(String[] args) {
        int[] arr = {5, 7, 2, 1, 8, 3, 4};
        display(arr);
        int idx = partition(arr, 0, arr.length-1);
        System.out.println("partition index: " + idx);
        display(arr);
    }
}
```

```
public static int partition(int[] arr, int si, int ei) {
    int pivot = arr[ei];
    int idx = si;
    for(int i=si; i<ei; i++){
        if(arr[i] < pivot){
            // swap  $\Rightarrow arr[i], arr[idx]$ 
            int temp = arr[i];
            arr[i] = arr[idx];
            arr[idx] = temp;

            idx++;
        }
        display(arr);
    }
}
```

```

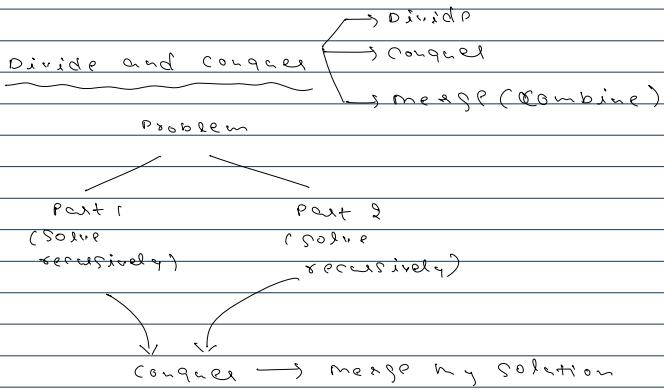
// swap => arr[idx], pivot(arr[e])
int temp = arr[e];
arr[e] = arr[idx];
arr[idx] = pivot;
return idx;
}

```

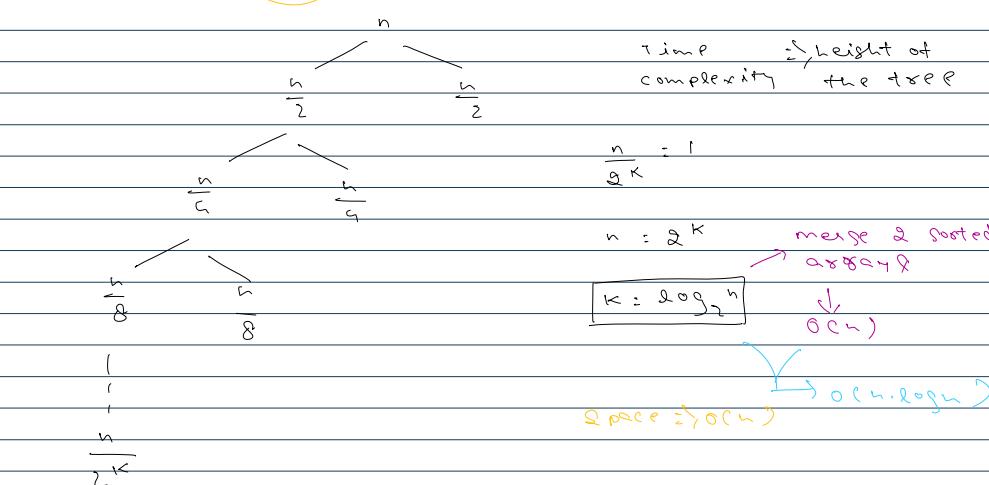
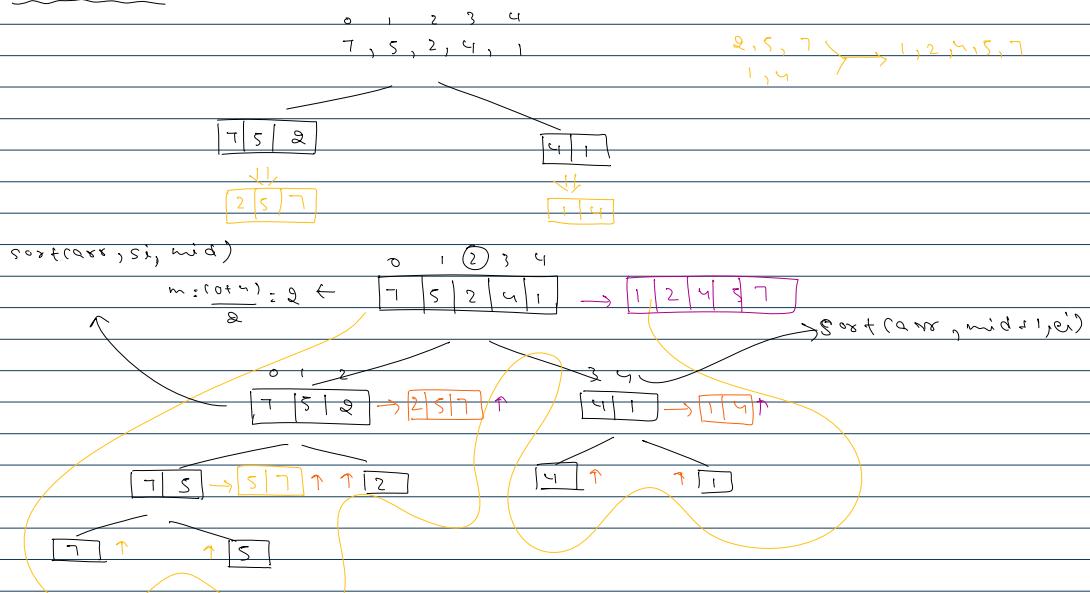
```

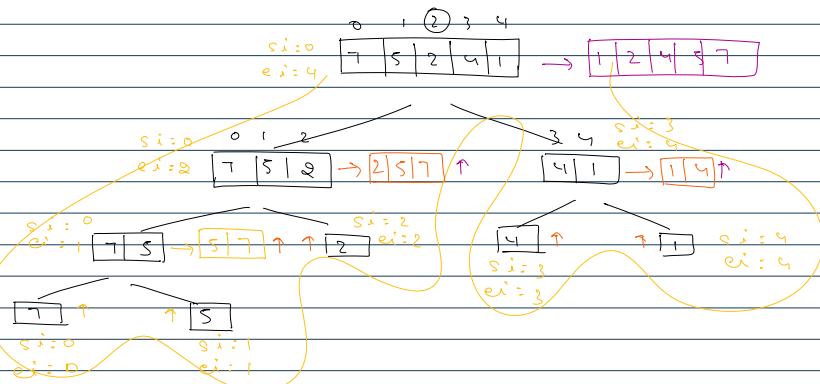
public static void display(int[] arr){
    for(int i=0; i<arr.length; i++){
        System.out.print(arr[i] + " ");
    }
    System.out.println();
}
}

```



### merge Sort





```

public class Main {
    public static void main(String[] args) {
        int[] arr = {5, 7, 2, 1, 8, 3, 4};
        display(arr);
        int[] sortedArray = sort(arr, 0, arr.length-1);
        display(sortedArray);
    }

    public static int[] sort(int[] arr, int si, int ei){
        if(si == ei){
            int[] bs = {arr[si]};
            return bs;
        }
        int mid = (si + ei)/2;
        int[] first = sort(arr, si, mid); //1st sorted array
        int[] second = sort(arr, mid+1, ei); //2nd sorted array
        return merge(first, second);
    }

    public static void display(int[] arr){
        for(int i=0; i<arr.length; i++){
            System.out.print(arr[i] + " ");
        }
        System.out.println();
    }

    public static int[] merge(int[] arr1, int[] arr2) {
        int n = arr1.length;
        int m = arr2.length;

        int[] arr = new int[n+m];

        int i = 0;
        int j = 0;
        int k = 0;

        while(i < n && j < m){
            if(arr1[i] < arr2[j]){
                arr[k] = arr1[i];
                i++;
            } else{
                arr[k] = arr2[j];
                j++;
            }
            k++;
        }

        while(i < n){
            arr[k] = arr1[i];
            i++;
            k++;
        }

        while(j < m){
            arr[k] = arr2[j];
            j++;
            k++;
        }
    }
}

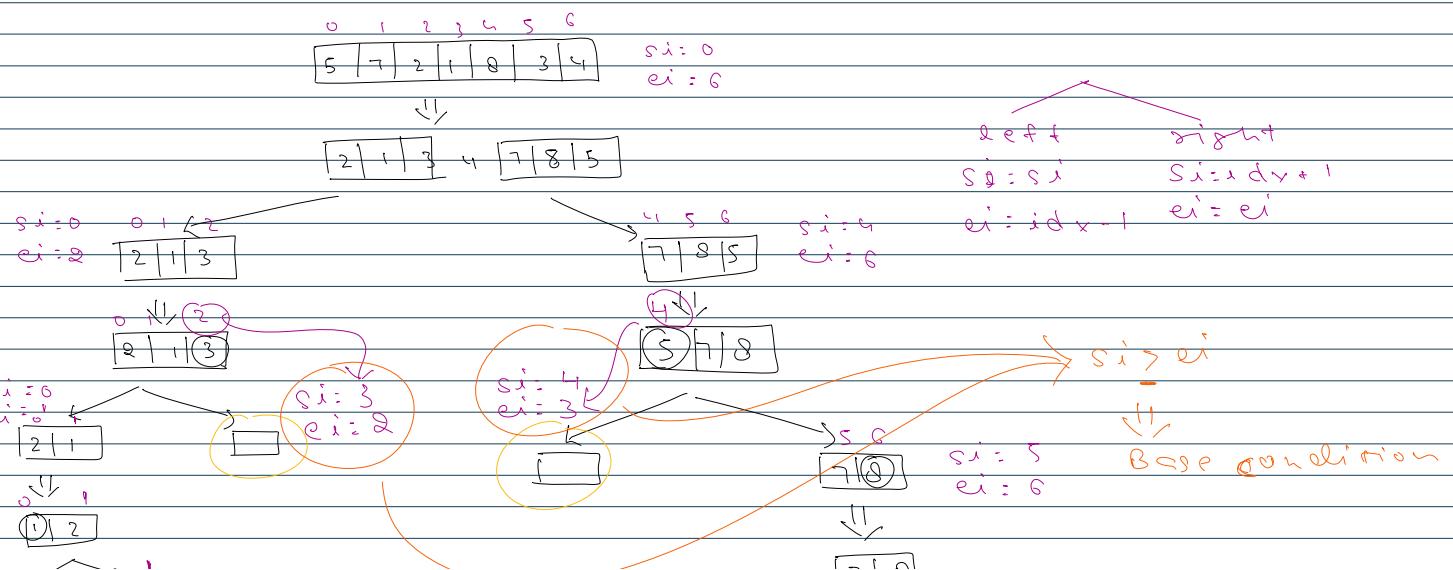
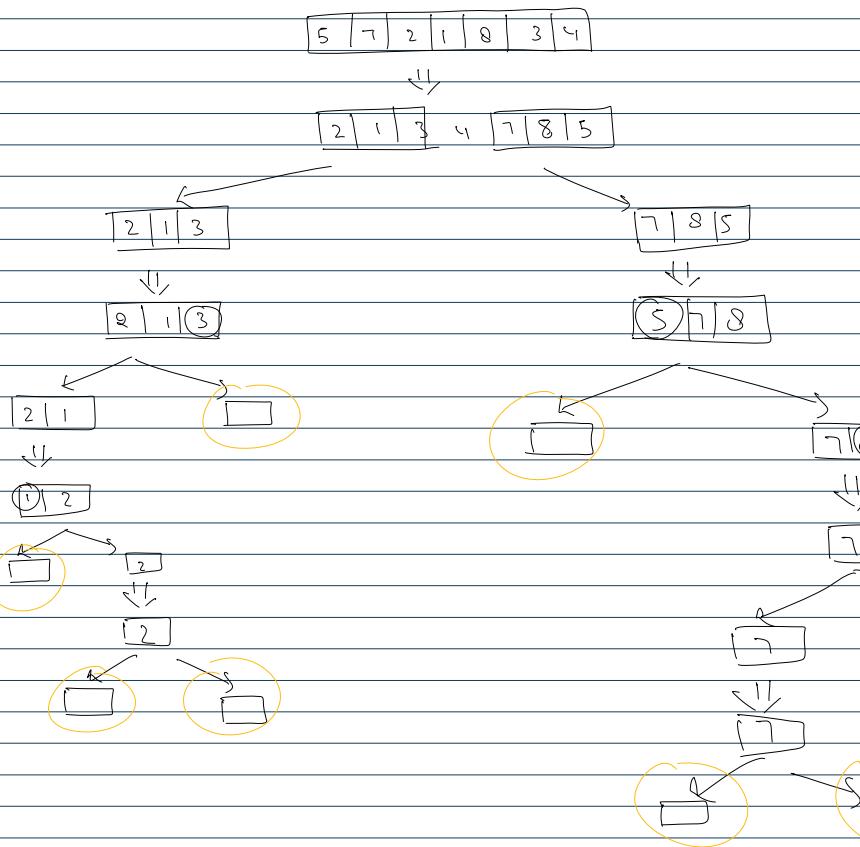
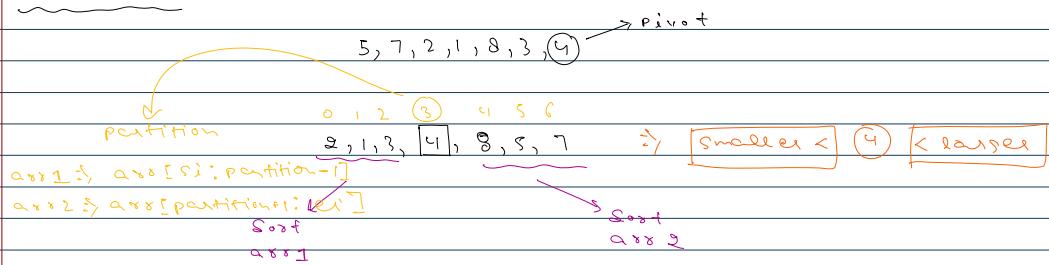
```

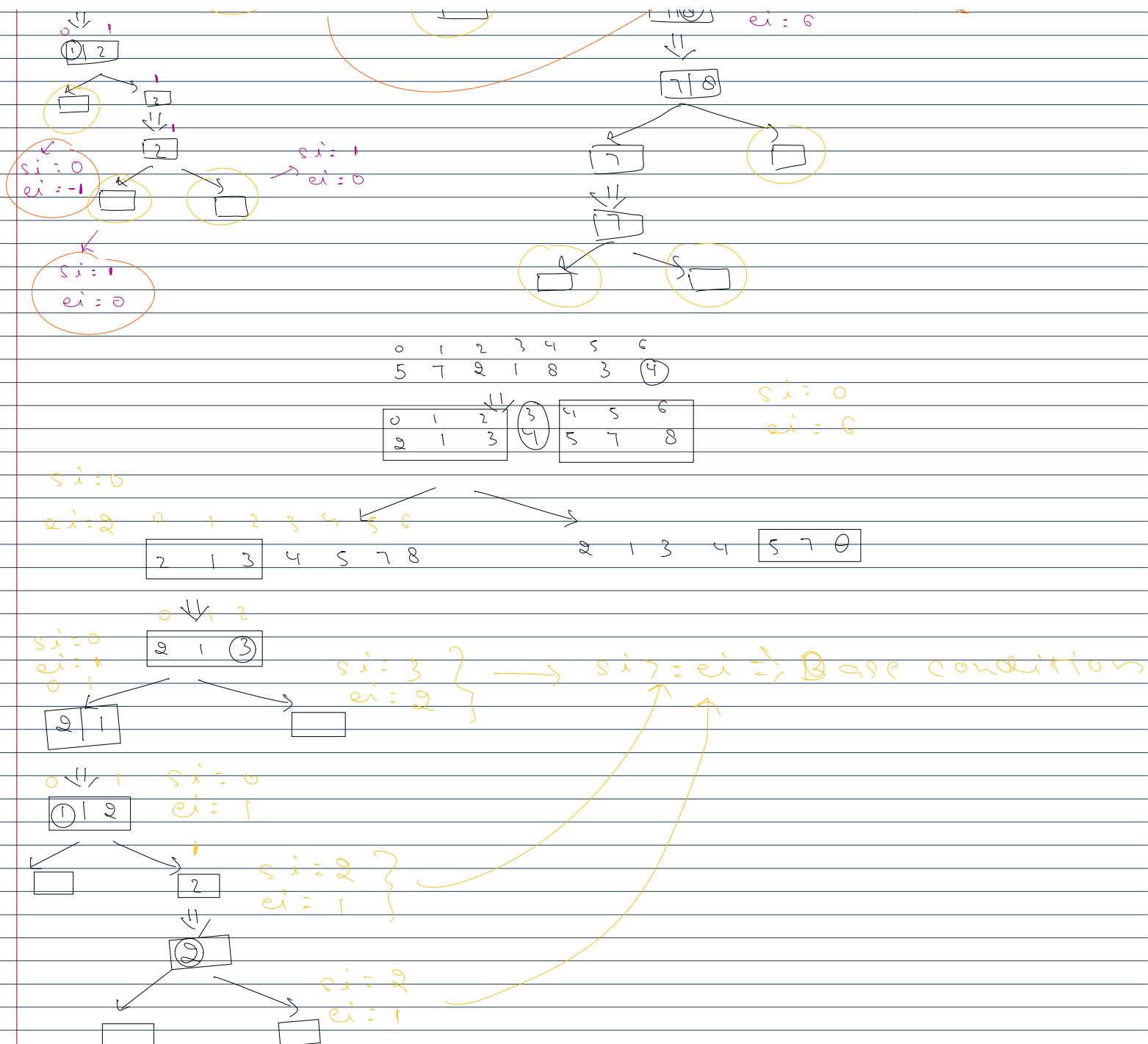
```

j++;
}
return arr;
}
}

```

### Quick Sort





```

public class Main {
    public static void main(String[] args) {
        int[] arr = {5, 7, 2, 1, 8, 3, 4};
        display(arr);
        sort(arr, 0, arr.length-1);
        display(arr);
    }

    public static void sort(int[] arr, int si, int ei){
        if(si >= ei){
            return;
        }
        int idx = partition(arr, si, ei);
        sort(arr, si, idx-1);
        sort(arr, idx+1, ei);
    }

    public static void display(int[] arr){
        for(int i=0; i<arr.length; i++){
            System.out.print(arr[i] + " ");
        }
    }
}

```

```

    }
    System.out.println();
}

public static int partition(int[] arr, int si, int ei){
    int pivot = arr[ei];
    int idx = si;
    for(int i=si; i<ei; i++){
        if(arr[i] < pivot){
            // swap => arr[i], arr[idx]
            int temp = arr[i];
            arr[i] = arr[idx];
            arr[idx] = temp;

            idx++;
        }
        display(arr);
    }
}

// swap => arr[idx], pivot(arr[ei])
int temp = arr[ei];
arr[ei] = arr[idx];
arr[idx] = pivot;
return idx;
}

```

|         | Worst      | Average    |
|---------|------------|------------|
| Space   | $n \log n$ | $n \log n$ |
| Inplace | X          | ✓          |
| Dom C   |            | O(nC)      |

Randomized quick sort

```

public class Main {
    public static void main(String[] args) {

        int low = 10;
        int high = 100;

        Random rn = new Random();
        for(int i=0; i<20; i++){
            int x = rn.nextInt(high-low+1) + low; // 91
            System.out.println(x);
        }
    }
}

```